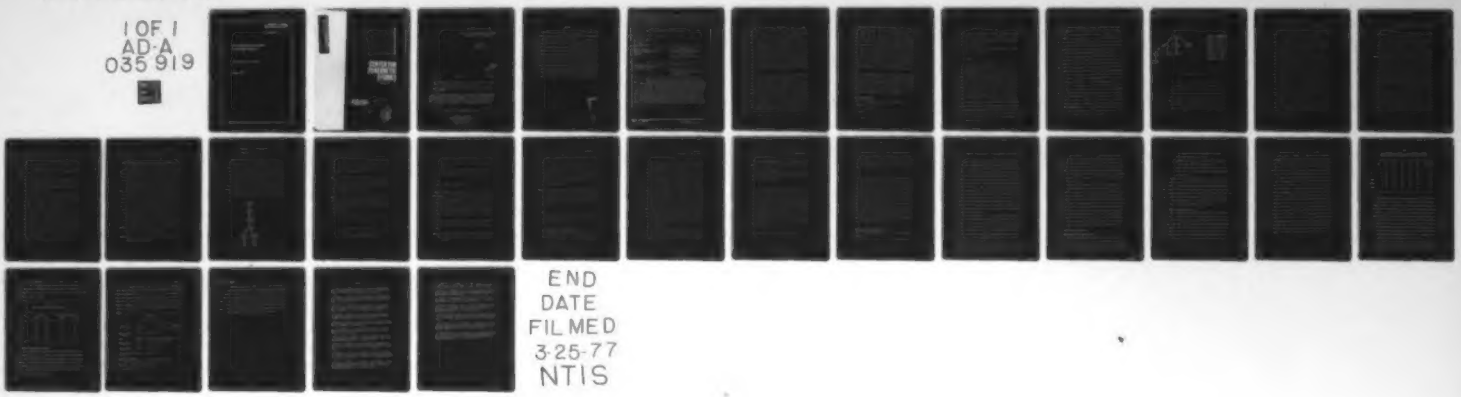
AD-A035 919 TEXAS UNIV AT AUSTIN CENTER FOR CYBERNETIC STUDIES F/G 12/2
THE ALTERNATING BASIS ALGORITHM FOR ASSIGNMENT PROBLEMS.(U)
JAN 77 R S BARR, F GLOVER, D KLINGMAN N00014-76-C-0616
UNCLASSIFIED CCS-263 NL

1 OF 1
AD-A
035 919



END
DATE
FILMED
3-25-77
NTIS

THE ALTERNATING BASIS ALGORITHM FOR ASSIGNMENT PROBLEMS

TEXAS UNIVERSITY AT AUSTIN

JANUARY 1977

CENTER FOR CYBERNETIC STUDIES

The University of Texas
Austin, Texas 78712

Research Report CCS 263

THE ALTERNATING BASIS ALGORITHM FOR ASSIGNMENT PROBLEMS

by

R. S. Barr*
F. Glover**
D. Klingman

January 1977

*Assistant Professor of Management Science and Computers, Southern Methodist University, Dallas, Texas 75275.

**Professor of Management Science, University of Colorado, Boulder, Colorado 80302.

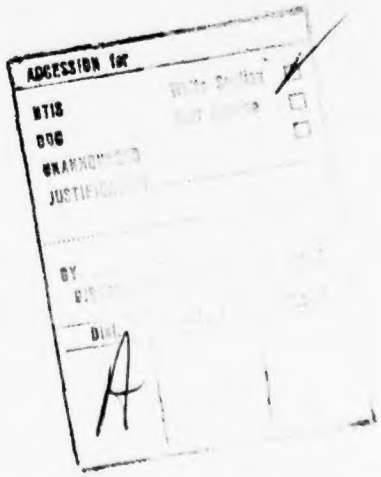
This research was partly supported by ONR Contract N00014-76-C-0383 with Decision Analysis and Research Institute and by Project NR047-021, ONR Contracts N00014-75-C-0616 and N00014-75-C-0569 with the Center for Cybernetic Studies, The University of Texas. Reproduction in whole or in part is permitted for any purpose of the United States Government.

CENTER FOR CYBERNETIC STUDIES

A. Charnes, Director
Business-Economics Building, 203E
The University of Texas
Austin, Texas 78712
(512) 471-1821

ABSTRACT

The purpose of this paper is to present a new primal extreme point algorithm for solving assignment problems which both circumvents and exploits degeneracy. The algorithm is based on the observation that the degeneracy difficulties of the simplex method result from the unnecessary inspection of alternative basis representations of the extreme points. This paper characterizes a subset Q of all bases that are capable of leading to an optimal solution to the problem if one exists. Using this characterization, an extreme point algorithm is developed which considers only those bases in Q. Computational results disclose that the new algorithm is substantially more efficient than previously developed primal and primal-dual extreme point ("simplex") methods for assignment problems.



Unclassified
Security Classification

DOCUMENT CONTROL DATA - R & D

Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified

1. ORIGINATING ACTIVITY (Corporate author)	2a. REPORT SECURITY CLASSIFICATION
Center for Cybernetic Studies The University of Texas	Unclassified 2b. GROUP

3. REPORT TITLE

The Alternating Basis Algorithm for Assignment Problems

4. DESCRIPTIVE NOTES (Type of report and, inclusive dates)

5. AUTHOR(S) (First name, middle initial, last name)

Richard S. Barr
Fred Glover
Darwin Klingman

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
January 1976	32	17

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
N00014-75-C-0616;0569 and N00014-76-C-0383 b. PROJECT NO. NR047-021 c. d.	Center for Cybernetic Studies Research Report CCS 263 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Office of Naval Research (Code 434) Washington, D.C.

13. ABSTRACT

The purpose of this paper is to present a new primal extreme point algorithm for solving assignment problems. The new algorithm both circumvents and exploits degeneracy. Degeneracy difficulties of the standard primal simplex method result from the unnecessary inspection of alternative basis representations of the extreme points. This paper characterizes a subset Q of all bases that are capable of leading to an optimal solution to the problem if one exists. The new algorithm is based on a special set of transition rules that make it possible to examine only bases in Q. The graph structure of Q imparts a number of computational advantages to the algorithm. A preliminary computer implementation, without refinement or investigation of alternative choice rules, is more efficient and uses less computer memory than the currently best implementations of out-of-kilter, primal-dual, and primal extreme point methods for assignment problems.

DD FORM 1 NOV 65 1473 (PAGE 1)
S/N 0101-807-6811

Unclassified
Security Classification

A-31408

1. INTRODUCTION

In spite of the notable gains in network solution techniques [1-5, 7-12, 14-16] since 1969, there is still a major area that remains to be explored. This is the area of designing appropriate techniques for accommodating and taking advantage of degeneracy within primal simplex-based algorithms. Computational testing has shown that approximately 90 percent of the pivots are degenerate for assignment and transshipment problems with more than 1000 nodes. Thus far, no computational schemes have been developed to respond to this situation, either by an effort to circumvent degenerate pivots or to make them judiciously. The need for such procedures has become increasingly urgent because practical problems are being formulated which involve several thousand nodes and hundreds of thousands of variables.

The purpose of this paper is to present a new primal extreme point algorithm for solving assignment problems which both circumvents and exploits degeneracy. The algorithm is based on the observation that the degeneracy difficulties of the simplex method result from the unnecessary inspection of alternative basis representations of the extreme points. The algorithm can be viewed as a variant of the independently developed method of Cunningham [6], specialized to take advantage of the bipartite structure of assignment networks. One of the principal features of this algorithm is a strong form of convergence that limits the number of degenerate steps in a far more powerful way than achieved by "lexicographic improvement," as for example, in

customary LP perturbation schemes.

Each basis examined by this algorithm is restricted to have a certain topology. We show that if an assignment problem has a feasible solution, then an optimal solution can be found by considering only bases of this type. The major mathematical differences between the AB algorithm and previous primal extreme point methods are (1) the rules of the algorithm automatically (without search) assure that all bases have the special topological structure, and bypass all other bases normally given consideration; (2) the algorithm is finitely convergent without reliance upon external techniques (such as lexicography or perturbation); and (3) in certain cases degenerate basis exchanges may be recognized prior to finding the representation of an incoming arc.

The AB algorithm also has special advantages for computer implementation. Specifically, the computer memory required to store the basis data is roughly half that required for other specialized algorithms for the assignment problem. In addition, the computational results reported in the final section demonstrate the notable efficiency of the AB algorithm.

2. BACKGROUND MATERIAL

An n x n assignment problem may be defined as:

$$\text{Minimize} \sum_{(i,j)\in A} c_{ij}\, x_{ij}$$

subject to:

$$\sum_{j\in\{j:(i,j)\in A\}} x_{ij} = 1, \quad i\in I=\{1,2,\ldots,n\}$$

$$\sum_{i\in\{i:(i,j)\in A\}} x_{ij} = 1, \quad j\in J=\{1,2,\ldots,n\}$$
$$x_{ij} \geq 0, \quad (i,j)\in A$$

where I is the set of originnodes, J is the set of destination nodes, A is the set of arcs, and c_{ij} is the cost of a unit flow on arc (i,j)(or in alternative terminology, of "assigning" origin node i to destination node j).

The dual of the assignment problem may be stated as:

Maximize $\sum_{i\in I} R_i + \sum_{j\in J} K_j$

subject to:

$$R_i + K_j \leq c_{ij}, \quad (i,j) \in A$$

where R_i and K_j are called the node potentials of the origin and destination nodes, respectively.

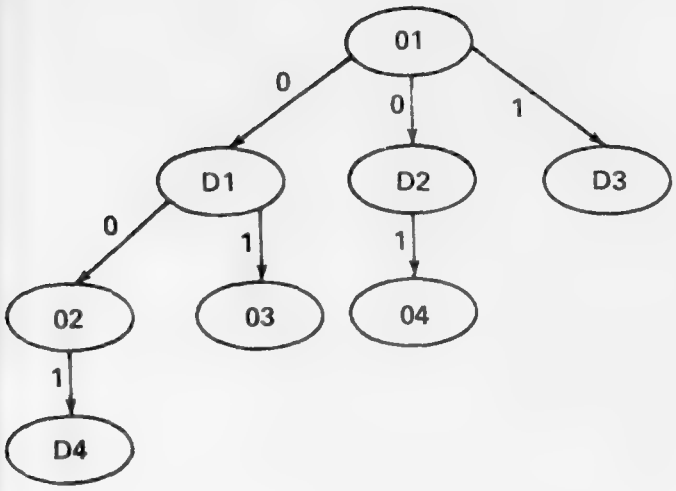
An understanding of the results of this paper relies on a familiarity with graphical interpretations of the assignment problem and how the primal simplex method has been applied to this problem. While these ideas are relatively direct they, unfortunately, are not succinctly itemized in any single reference and will be summarized in this section for completeness.

As our preceding terminology suggests, the assignment problem may be represented as a bipartite graph, consisting of a set of origin nodes with unit supplies and a set of destination nodes with unit demands. Directed arcs from origin nodes to destination nodes accomodate the transmission of flow and incur a cost if flow exists. The objective is to determine a set of arc flows which satisfies the supply and demand requirements at minimum total cost.

The bases of an extreme point (simplex) method for solving an nxn assignment problem correspond to spanning trees with 2n-1 arcs. A basic solution assigns exactly n of the basic arcs a flow value of one and the other n-1 arcs a flow value of zero (all nonbasic arcs receive flows of 0.). Therefore each basis solution is highly degenerate (i.e. contains a large number of zero flows on basic arcs). This often causes a primal extreme point method to examine several alternative bases for the same extreme point before moving to an adjacent extreme point.

In the graphical representation approach, the bases of the simplex method of assignment problems are normally kept as rooted trees [7,9,11,12,16]. Conceptually, the root node may be thought of as the highest node in the tree with all of the other nodes hanging below it on directed paths leading downward from the root. Those nodes in the unique path from the root to any given node i are called the ancestors of node i, and the immediate ancestor of node i is called its predecessor.

Figure 1 illustrates a rooted basis tree, the predecessors of the nodes, and the basic flow values, for a 4-4 assignment problem. Notationally, Oi denotes the ith origin node and Dj denotes the jth destination node. The number beside each link (arc) in the basis tree indicates the flow on this arc imparted by the basic solution. Predecessors of nodes are identified in the NODE/PREDECESSOR array. For example, as seen from this array, the predecessor of origin node 2 is destination node 1. The root of the tree is node 01 and has no predecessor.



NODE	PREDECESSOR
01	None
02	D1
03	D1
04	D2
D1	01
D2	01
D3	01
D4	02

Figure 1- Rooted Basis Tree

It is important to note the direction of the links in Figure 1 correspond to the orientation induced by the predecessor ordering and do not necessarily correspond to the direction of the basis arcs in the assignment problem. However, the direction of the basic arcs are known from the bipartite property of the assignment problem; i.e., all problem arcs lead from origin nodes to destination nodes.

In subsequent sections the term O-D link and D-O link will be used to refer to links in a rooted basis tree that

are directed from an origin node to a destination node and vice versa, respectively, according to the orientation imparted to the basic arcs by the predecessor indexing. For example, in Figure 1, O2-D4 is a O-D link while D1-O2 is a D-O link. Additionally, basic arcs with a flow of one or zero will be referred to as 1-links and 0-links, respectively.

We will now briefly review the fundamental pivot step of the simplex method in the graphical setting. Assume that a feasible starting basis has been determined and is represented as a rooted tree. To evaluate the nonbasic arcs to determine whether any of them "price out" profitably, and therefore are candidates to enter the basis, it is necessary to determine values for the dual variables R_i, $i \in I$, and K_j, $j \in J$, which satisfy complementary slackness; i.e., which yield $R_i + K_j = c_{ij}$ for each basic arc.

There is a unique dual variable associated with each node in the basis tree. For this reason the dual variables--or their values--are often referred to as node potentials. Because of redundancy, in the defining equations of the assignment problem (and in network problems generally), one node potential value may be specified arbitrarily. The root node is customarily selected for this purpose and assigned a potential value of zero, whereupon the potentials of the other nodes are immediately determined in a cascading fashion by moving down the tree and identifying the value for each node from its predecessor using the equation $R_i + K_j = c_{ij}$.

Highly efficient labeling procedures for traversing the tree to initialize and update these node potential values are described in [9,11,12,16].

A feasible basic solution is optimal when the nonbasic arcs all satisfy the dual feasibility condition $R_i + K_j \leq c_{ij}$. If the problem is not optimal, then an arc whose dual constraint is violated (i.e., for which $R_i + K_j > c_{ij}$) is selected to enter the basis. The arc to leave the basis is determined by: (1) finding the unique path in the basis tree, called the basis equivalent path, which connects the two nodes of the entering arc, and (2) isolating a blocking arc in this path whose flow goes to zero ahead of (or at least as soon as) any others as a result of increasing the flow on the entering arc. In the basis equivalent path all arcs an even number of links away from the entering arc are called even arcs, and all arcs an odd number of links away are called odd arcs. An increase in the flow of the incoming arc causes a corresponding increase in the flow of all even arcs and a corresponding decrease in the flow of all odd arcs. Thus, if an odd arc already has a 0 flow, then such an arc qualifies as a blocking arc and the incoming arc cannot be assigned a positive flow value.

To illustrate, assume that the starting basis is the one given in Figure 1 and the entering arc is (O4,D4). The basis equivalent path for (O4,D4) is D4-O2-D1-O1-D2-O4. (Note that this path can be easily determined by tracing the predecessors of O4 and D4 to their point of intersection [7,9,11,12]. As

flow is increased on the entering arc the flow on the arc (O1,D1), which is an odd number of links away, must be decreased. However, its flow is already zero, and hence (O1,D1) qualifies as a blocking arc. When arc (O4,D4) is brought into the basis, arc (O1,D1) must be dropped (since there are no other blocking arcs in this case). In addition, the pivot (or basis exchange) is degenerate since no flow increase occurs.

Once the entering and leaving arcs are known, the basis exchange is completed simply by updating the flow values on the basis equivalent path and determining new node potentials for the new basis tree.

Only a subset of the node potentials change during a pivot and these can be updated rather than being determined from scratch. This fact will play a crucial role in proving convergence of the algorithm to be developed.

To update the node potentials, assume that the nonbasic arc (p,q) is to enter into the basis and the basic arc (r,s) is to leave the basis. If arc (r,s) is deleted from the basis (before adding arc (p,q)), two subtrees are formed. each containing one of the two nodes of the incoming arc (p,q). Let K denote the subtree which does not contain the root node of the full basis. The node potentials for the new basis may be obtained by updating only those potentials of the nodes in K, as follows [9]. If p is in K, subtract $\delta = R_p + K_q - c_{pq} > 0$ from the potential of each origin node

in K and add δ to the potential of each destination node in K. Otherwise, q is in K and $-\delta$ is used in the above operations.

3. ALTERNATING PATH BASIS DEFINITION AND PROPERTIES

The new alternating basis (AB) algorithm for assignment problems developed in this section is similar to the primal simplex method as described above. Its major mathematical distinction is that it does not consider all feasible bases to be candidates for progressing to an optimal basis. That is, the simplex method allows a feasible spanning tree of any structure whatsoever to be included in the set of those that are eligible for consideration as "improving bases" along the path to optimality. However, it will be shown that if an assignment problem has an optimal solution then it also has an optimal solution with a unique basis tree structure, dubbed the alternating path (AP) structure. Furthermore, we will show that it is possible to restrict attention at each step to bases with this structure. In particular, the AB algorithm is a procedure designed to exploit the properties of the AP basis structure in a manner that substantially reduces the impact of degeneracy, the number of arithmetic operations, and the data storage locations required to solve the assignment problem.

Definition: A rooted basis tree for an assignment problem is an alternating path (AP) basis if

1. The root node is an origin node.
2. All 1-links are O-D links.
3. All 0-links are D-O links.

An example of an AP basis is shown in Figure 2.

The "alternating path" designation is applied because every path from a node to any ancestor node in the tree, or vica versa, is an alternating path of 1-links and 0-links. We will chiefly be concerned with paths from nodes to their ancestors (as would be traced along a succession of predecessors). A path that begins at an origin node and ends at an ancestor destination node will be called "0-AP" because it begins and ends with a 0-link. Similarily, a path that begins at a destination node and ends at an ancestor origin node will be called "1-AP" because it begins and ends with a 1-link.

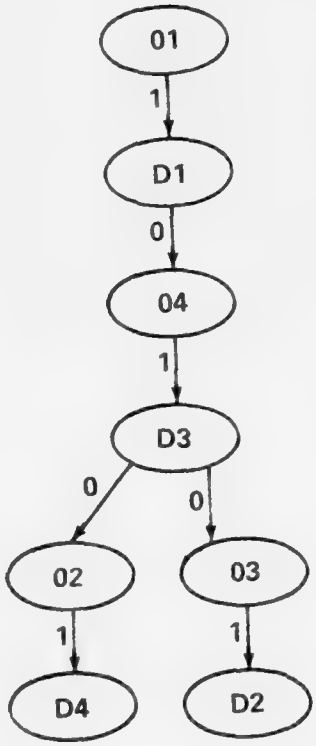


Figure -2-An AP basis structure for a 4x4 assignment problem

Remark 1: The 1-links of any feasible assignment solution can be augmented by 0-links to create an AP basis (e.g., by adding 0-links from destination nodes to origin nodes in any fashion so that every origin node except the root node has exactly one entering 0-link.) Note if the arcs corresponding to the added 0-links do not exist in the assignment problem then a large (big M) cost is assigned such links.

Remark 2: There are many assignment bases for a given feasible solution that are not AP bases. (For example, any basis that has more than one 0-link incident to an origin node is not an AP basis regardless of the origin node chosen as the root. Figure 1 is an example of such a basis.)

Remark 3: An artificially feasible AP basis may always be constructed for an nxn assignment problem by assuming that arcs exist from each origin node to all destination nodes where the non-admissible (artificial) arcs have a "big M" cost. The procedure is as follows.

Initially set $J'=\emptyset$, $B=\emptyset$, and $i=1$. Go to step 1.

1. Let r be a destination node such that $c_{ir} = \min_{\substack{j\in J-J' \\ (i,j)\notin B}} c_{ij}$

   Set $B=\{(i,r)\} \cup B$, $x_{ir}= 1$, and $J'=J' \cup \{r\}$.

2. If $i \neq n$, set $i= i+1$ and go to step 1. Otherwise set $i=2$, $J'=\emptyset$, $B'=B$, and go to step 3.

3. Let r be a destination node such that $c_{ir} = \min_{\substack{j \in J - J' \\ (i,r) \notin B'}} c_{ij}$

   Set $B = \{(i,r)\} \cup B$, $x_{ir} = 0$, and $J' = J' \cup \{j:(i,j) \in B'\}$

4. If $i \neq n$, set $i = i+1$ and go to step 3. Otherwise go to step 5.

5. Using B, create a spanning tree rooted at node 1. The resulting spanning tree will be an AP basis.

Proof:

The remark follows by construction.

Definition: Relative to any AP basis, a nonbasic arc is called a downward arc if it connects a destination node to an ancestor origin node, an upward arc if it connects an origin node to an ancestor destination node. An arc that connects an origin node and a destination node that do not have either of these ancestral relationships is called a cross arc. (Note that these are the only three possibilities for a nonbasic arc in a bipartite network.)

The next two remarks point out some important properties that can be exploited when applying the simplex method to an AP basis.

Remark 4: When the simplex method is applied to an AP basis, a pivot is nondegenerate if and only if the entering nonbasic arc is a downward arc.

Proof:

The remark relies on the fact that a nondegenerate pivot causes the flows on the basis equivalent path to decrease and increase in a strictly alternating fashion on the odd and even links. The "if" part of the remark then follows by observing that a downward arc is 1-AP. The "only if" part of the remark follows from two observations, first that an upward arc is 0-AP, and second that a cross arc has a 0-link above the origin node incident to the entering arc (and this arc is contained in the basis equivalent path adjacent to one of the nodes of the entering arc).

Remark 5: When the simplex method is applied to an AP basis, the pivot can be carried out to give a new AP basis for any entering nonbasic arc simply by dropping the unique link in the basis equivalent path attached to the origin node of the entering arc.

Proof:

The remark follows by observing that an AP basis results if a rooted tree is constructed with its root node equal to the root node of the old AP basis.

Alternating Basis (AB) Algorithm

On the basis of the preceding remarks the rules of the AB algorithm can be stated in an extremely simple fashion.

1. Select any feasible AP basis for the assignment problem (e.g. using Remark 3).

2. Successively apply the simplex pivot step keeping the root node fixed and picking the link to leave according to Remark 5.

By means of these rules, the foregoing observations imply that the AB algorithm will proceed through a sequence of AP bases, bypassing all other basis structures. Further, these remarks show that a "next" AP basis is always accessible to a given AP basis, so that the method will not be compelled to stop prematurely without being able to carry out a pivot before the optimality (dual feasibility) criteria are satisfied. The issue to be resolved then, is whether the method may progress through a closed circle of AP bases without breaking out, and thus fail to converge. We will show that this cannot happen, and that in fact the AB algorithm is finitely converging without any reliance upon "external" techniques such as perturbation, as is the ordinary simplex method. Moreover we will show that the form of convergence of the AB algorithm has a particularly strong character, in which origin node potentials and destination node potentials each change in a uniform direction throughout any sequence of degenerate pivots.

These results do not require any restrictions on the choice of the incoming variable. For example it is not necessary to cull through pivot possibilities in an attempt to find degenerate pivot candidates. We now complete their foundations as follows.

Lemma: A basis exchange with the AB algorithm gives rise to a new AP basis in which the new node potentials satisfy the following properties:

a) For a nondegenerate pivot: The changed origin node potentials strictly increase and the changed destination node potentials strictly decrease.

b) For a degenerate pivot: The changed origin node potentials strictly decrease and the changed destination node potentials strictly increase.

Proof:

As already discussed, the node potential values that change may be restricted to those associated with the subtree K. By this procedure, if subtree K contains the origin node of the entering arc then all the origin node potentials in K are decreased and all destination node potentials in K are increased. The reverse is true if the destination node of the entering arc is in subtree K. The lemma then follows from Remarks 4 and 5, which imply that subtree K always contains the destination node of the entering arc for a nondegenerate pivot and the origin node of the entering arc for a degenerate pivot.

Our main result may be stated as follows.

Theorem:

The AB algorithm will obtain an optimal solution (or determine that the problem is infeasible) in a finite number of pivots, regardless of which dual infeasible arc is chosen to be the entering arc, and without any reliance on perturbation or lexicographic orderings.

Proof:

It is sufficient to show that the number of degenerate pivots that occur between any two nondegenerate pivots must be finite. This follows from the second half of the lemma. Note that the node potential assigned to the root node never changes when the node potentials in subtree K are updated. Thus given the constant node potential for the root, the other node potentials are uniquely determined for each successive basis (regardless of the procedure by which they are generated), and the uniform decrease of origin node potentials and the uniform increase of destination node potentials (for the potentials that change) implies that no basis can ever repeat during an uninterrupted sequence of degenerate pivots. This completes the proof.

4. COMPUTATIONAL CONSIDERATIONS

Some of the unique computational features of the AB method include:

a) It explicitly bypasses all "non-AP" basis solutions

without requiring any imbedded search procedure or computational tests.

b) It allows degenerate pivots to be recognized and performed without computing the representation of the entering arc. This can be accomplished by using the "cardinality function" proposed by Srinivasan and Thompson [17] which indicates the number of nodes in the subtree below a given node. In particular, upward arcs and some cross arcs can be detected simply by comparing the cardinality function values of the nodes associated with the entering arc. That is, denote the cardinality function by f and the entering arc by (p,q). If $f(p)<f(q)$ then arc (p,q) is either an upward arc or a cross arc. In either case the pivot is degenerate and no flow updating is required. Remark 5, furthermore directly specifies the link to leave the basis. Thus a degenerate pivot simply involves checking the cardinality function, inserting and deleting the appropriate links, and updating the node potential values.

c) Similar streamlining can be achieved for all other pivots. Specifically, if $f(q)<f(p)$ then the appropriate step is to find the unique node z on the path from q to the root node such that $f(z)\geq f(p)$. If $z\neq p$ then arc (p,q) is a cross arc and thus the pivot may be executed as before. If $z=p$ then the arc (p,q) is a downward arc and the pivot is nondegenerate. Note that it is only in the case of a nondegenerate pivot that the entire basis equivalent path of the entering arc is traversed. Thus it is only in this case that the complete representation of the entering arc is computed. This is, of course, substantially different than for the standard simplex method.

Steps b and c above can also be accomplished by using the "distance function" of [17] which indicates the number of links in the basis tree between a given node and the root node. In particular, the distance function may be used in place of the cardinality function by reversing all the above inequalities.

d) Flow values on the basis links never have to be checked to determine the type of pivot. In fact it is not necessary to store the flows at all.

e) The structural property of an AP basis whereby all 1-links are 0-D links allows the basis tree to be easily stored in a "collapsed" form and makes it unnecessary to store and to update any flow values. Specifically, the two nodes and connecting arc associated with an 0-D link may be treated as a single collapsed node in storing and updating the AB basis. Using this technique, the basis computer storage requirement of the AB algorithm is roughly halved. Further the compression reduces the depth of the basis tree by approximately half. Thus less checking is required to find node z discussed in c). Also since there are only half as many nodes in the collapsed tree, this reduces the work required to update node potentials by fifty percent.

5. COMPUTATIONAL COMPARISON

5.1 Development of a Computer Code

We have embodies the AB algorithm in an in-core computer code entitled AP-AB. This code is written in FORTRAN, using a modular design with several subroutines to perform the various updating operations. The code was given this form for the following reasons:

a) This modular approach simplifies testing of different updating procedures.

b) Minimal recoding is required to fit different machine and compiler conventions, and

c) unbiased comparisons can be made with codes that have not been "customized" to a particular machine or compiler.

One disadvantage of this coding format, of course, is that the reported times are conservative, since programs which have been "tuned" to a particular operating environment execute substantially faster. To further facilitate unbiased comparisons with other primal extreme point, out-of-kilter and primal-dual algorithms, we used the same pivoting procedures as described in [8, 10]. The code also uses the predecessor [7], thread [12], and distance [17] functions to maintain and update the basis data.

5.2 Computational Comparisons

The following computer codes were obtained for comparative evaluations: SUPERK [3], one of the fastest out-of-kilter codes; PNET-I [8] the most widely-used primal simplex capacitated transshipment code; ARC-II [2] the fastest primal simplex capacitated transshipment code; and SUPERT-2 [1], the fastest uncapacitated transportation code. All of the codes are in-core codes, i.e., the program and all of the problem data simultaneously reside in fast-access memory. They are all coded in FORTRAN and none of them (including the special purpose primal codes) have been

optimized for a particular compiler. It is important to note that all the codes except for SUPERT-2 and AP-AB codes are designed to solve capacitated transshipment problems and are not specifically designed to exploit the special structure of assignment problems. Further, SUPERT-2 is designed to solve any uncapacitated transportation problem. All of the problems were solved on the CDC 6600 at the University of Texas Computation Center using the RUN compiler. The computer jobs were executed during periods when the machine load was approximately the same, and all solution times are exclusive of input and output. The total time spent solving the problem was recorded by calling a Real Time Clock upon starting to solve the problem and again when the solution was obtained.

In addition, we sought to compare the AP-AB code with a primal-dual code PD-AAL developed by Decision Systems Associates (DS) (reported by Hatch [13]). Since the DSA code is proprietary, we could not obtain a copy of it for comparison. However, we have solved the same test problems as solved by the DSA code using the same machine (a CDC-6600). These results are contained in Table I. It is important to note that it is our understanding that the DSA code, in contrast to the AP-AB code and the other codes tested, is fully optimized to exploit the special hardware features of the CDC-6600. This type of specialization could increase the performance of all the other codes by a factor of 2 or 3.

Table 1

TOTAL SOLUTION TIMES ON 200 X 200 ASSIGNMENT PROBLEMS
(IN SECONDS) ON A CDC 6600 WITH A COST RANGE OF 1-100

No. of Arcs	1500	2250	3000	3750	4500	Sum of Times
AP-AB	.97	1.12	1.48	1.61	1.68	6.86
ARC II	1.45	1.95	2.47	2.67	3.13	11.67
PD-AAL	1.63	1.14	1.89	1.29	1.80	7.75
PNET-I	2.31	3.71	3.47	3.44	4.79	17.72
SUPERK	6.44	6.47	7.25	6.95	7.56	34.67
SUPERT-2	1.26	1.57	1.98	2.17	2.53	9.51

A noteworthy feature of the computational results is that AP-AB, PD-AAL, SUPERT-2, and ARC-II are, in this order, superior to the other codes for assignment problems. Based on the sum of the solution times, AP-AB is roughly fifteen percent faster than its closest competitor (the machine-tuned code) and is roughly fifty percent faster than its next closest competitor. Further, the computational results of Table II indicate that the AB algorithm reduces the number of pivots by 25% over the most efficient version of the primal simplex algorithm that does not make recourse to the AB basis structure. Additionally, the results indicate that the AB algorithm reduces both the number of degenerate pivots and the number of nondegenerate pivots on the denser problems. Moreover, relative reduction in the nondegenerate pivots (that is, the number of extreme points visited) increases as the number of arcs increase.

These results also indicate that this first implementation of the AB algorithm is 1 1/2 times more efficient than the fastest uncapacitated primal simplex transportation code SUPERT-2. Historically, it has always been possible

to improve the solution speed of the first implementation of an algorithm by a factor of 2 or 3. Coupling this with the fact that the start and pivot rules of AP-AB have not been computationally investigated, it would appear by conservative estimate that the AB algorithm is likely twice as fast as other algorithms for solving assignment problems.

TABLE II

200 X 200 ASSIGNMENT PROBLEM STATISTICS

No. of Arcs	Total Pivots		Total Nondegenerate Pivots	
	Supert-2	AP-AB	Supert-2	AP-AB
1500	1420	1248	47	59
2250	1979	1498	62	73
3000	2150	1512	71	73
3750	1736	1356	63	52
4500	1944	1586	80	70

5.3 Memory Requirements of the Codes

Table III indicates the number of origin, destination, and arc length arrays required in each of the codes tested for solving assignment problems except for the PD-AAL code. The storage requirements of this code were not available. It should be noted that memory requirements of all of the codes tested were quite close (within 1500 words) excluding the array requirements. Thus, the important factor in comparing the codes is the number of origin, destination, and arc length arrays.

Looking at Table III and keeping in mind that any meaningful problem has to have more arcs than nodes, it is clear that the AB, and primal simplex, codes have a distinct advantage (in terms of memory requirements) over all of the other codes. Further, this advantage greatly increases as the number of arcs increase. It is important to note that the AB based code, AP-AB, requires the least amount of memory of all the codes making it the most suitable procedure for large scale problems both in terms of efficiency and memory.

Table III

Code Specifications

Developer	Name	Type	Number of Arrays
1. Barr, Glover Klingman	AP-AB	AB Algorithm	6N + 2A
2. Barr	SUPERT-2	Primal Simplex Transportation	10N + 2A
3. Barr, Glover, Klingman	ARC-II	Primal Simplex Network	14N + 2A
4. Barr, Glover, Klingman	SUPERK	Out-of-kilter	8N + 9A
5. Decision System Associates	PD-AAL	Primal-Dual	Not available
6. Glover, Karney, Klingman, Stutz	PNET-I	Primal Simplex Network	12N + 2A

N = Number of origins or destinations

A = Number of arcs

ACKNOWLEDGEMENTS

This research was partly supported by Project NRO74-146, ONR Contract N00014-76-C-0383 with Decision Analysis and Research Institute and by Project NRO47-021, ONR Contracts N00014-75-C-0616 and N00017-75-C-0569 with the Center for Cybernetic Studies, The University of Texas. The authors wish to acknowledge the Cooperation of the staff of The University of Texas Computation Center, The University of Texas Business School Computation Center, and the Southern Methodist University Computation Center.

REFERENCES

1. R. S. Barr, "Streamlining primal simplex transportation codes," Research Report to appear, Center for Cybernetic Studies, University of Texas, Austin, TX 78712.

2. R. S. Barr, F. Glover, and D. Klingman, "Enhancements to spanning tree labeling procedures for network optimization," Report 262, Center for Cybernetic Studies, University of Texas, Austin, TX 78712

3. R. S. Barr, F. Glover, and D. Klingman, "An improved version of the out-of-kilter method and a comparative study of computer codes," Mathematical Programming, 7, 1, 60-87 (1974).

4. G. Bradley, G. Brown, and G. Graves, "A comparison of storage structure for primal network codes," presented at ORSA/TIMS conference, Chicago, April 1975.

5. G. Bradley, G. Brown, G. Graves, "Tailoring primal network codes to classes of problems with common structure," ORSA/TIMS conference, Las Vegas (1975).

6. W. H. Cunningham, "A network simplex method," Technical Report No. 207, Dept. of Mathematical Sciences, Johns Hopkins University (1974).

7. F. Glover and D. Klingman, "Locating stepping-stone paths in distribution problems via the predecessor index method," Transportation Science, 4, 220-226 (1970).

8. F. Glover, D. Karney, and D. Klingman, "Implementation and computational study on start procedures and basis change criteria for a primal network code," Networks, 4, 3, 191-212 (1974).

9. F. Glover, D. Karney, D. Klingman, "Augmented predecessor index method for location stepping stone paths and assigning dual prices in distribution problems, Transportation Science, 6, 1, 171-181 (1972).

10. F. Glover, D. Karney, D. Klingman, and A. Napier, "A computational study on start procedures, basis change criteria, and solution algorithms for transportation problems," Management Science, 20, 5, 793-819 (1974).

11. F. Glover and D. Klingman, "Improved labeling of L.P. bases in networks," Research Report CS 218, Center for Cybernetic Studies, University of Texas, Austin, TX 78712 (1974).

12. F. Glover, D. Klingman, and J. Stutz, "Augmented threaded index method," Canadian Journal of Operational Research and Information Processing, 12, 3, 293-298 (1974).

13. R. S. Hatch, "Optimization strategies for large scale assignment and transportation type problems," ORSA/TIMS conference, San Juan, Puerto Rico (1974).

14. D. Klingman, A. Napier, and J. Stutz, "NETGEN-A program for generating large scale (un)capacitated assignment, transportation, and minimum cost flow network problems," Management Science, 20, 5, 814-822 (1974).

15. J. Mulvey, "Column weighting factors and other enhancements to the augmented threaded index method for network optimization," Joint ORSA/TIMS conference, San Juan, Puerto Rico (1974).

16. V. Srinivasan and G. L. Thompson, "Benefit-cost analysis of coding techniques for the primal transportation algorithm," Journal of the Association of Computing Machines, 20, 194-213 (1973).

17. V. Srinivasan and G. L. Thompson, "Accelerated Algorithms for labeling and relabeling of trees with application for distribution problems," Journal of the Association of Computing Machines, 19, 4, 712-726 (1972).